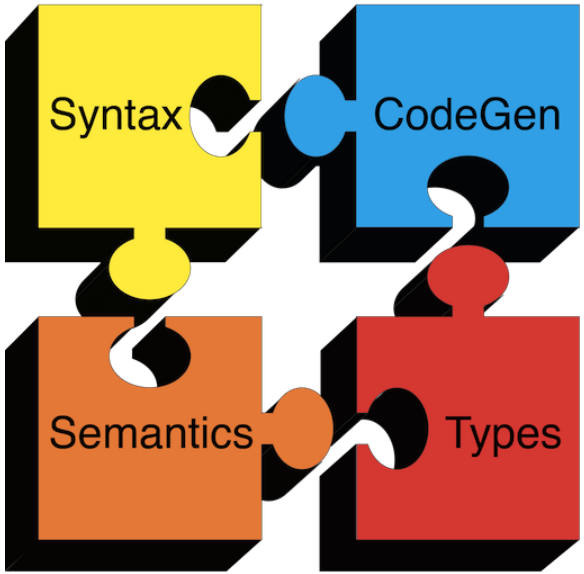

Compilers

Operational Semantics

Operational Semantics

- Once again we introduce a formal notation
- Logical rules of inference, as in type checking

Operational Semantics

- Recall the typing judgment

  Context $\vdash$ e : C

  In the given context, expression e has type C

- We use something similar for evaluation

  Context $\vdash$ e : v

  In the given context, expression e evaluates to value v

Operational Semantics

$$\frac{\text{Context} \vdash e_1 : 5 \qquad \text{Context} \vdash e_2 : 7}{\text{Context} \vdash e_1 + e_2 : 12}$$

Operational Semantics

- Consider the evaluation of y ← x + 1

- We track variables and their values with:
  - An environment : *where* in memory a variable is
  - A store : *what* is in the memory

- A variable environment maps variables to locations
  - Keeps track of which variables are in scope
  - Tells us where those variables are

$$E = [a : l_1, b : l_2]$$

- A store maps memory locations to values

$$S = [l_1 \rightarrow 5, l_2 \rightarrow 7]$$

- $S' = S[12/l_1]$ defines a store S' such that

$$S'(l_1) = 12 \quad \text{and} \quad S'(l) = S(l) \text{ if } l \neq l_1$$

- Cool values are objects
  - All objects are instances of some class

- $X(a_1 = l_1, \ldots, a_n = l_n)$ is a Cool object where
  - X is the class of the object
  - a_i are the attributes (including inherited ones)
  - l_i is the location where the value of a_i is stored

- Special cases (classes without attributes)

  Int(5) the integer 5

  Bool(true) the boolean true

  String(4, “Cool”) the string “Cool” of length 4

- There is a special value void of type Object
  - No operations can be performed on it
  - Except for the test isvoid
  - Concrete implementations might use NULL here

- The evaluation judgment is

  so, E, S ⊢ e : v, S’

  - Given so the current value of self
  - And E the current variable environment
  - And S the current store
  - If the evaluation of e terminates then
  - The value of e is v
  - And the new store is S’

Operational Semantics

- "Result" of evaluation is a value and a store
  - New store models the side-effects
- Some things don't change
  - The variable environment
  - The value of self
  - The operational semantics allows for non-terminating evaluations